

FIG. 1

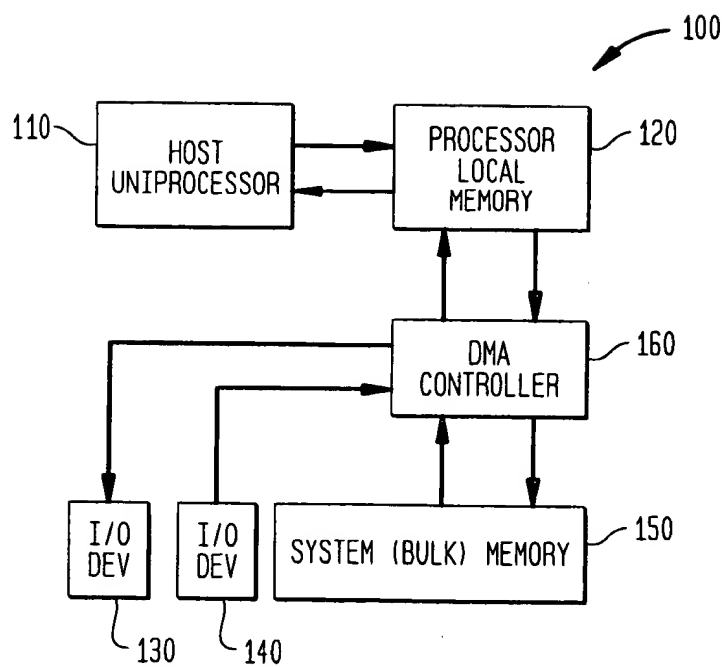


FIG. 2

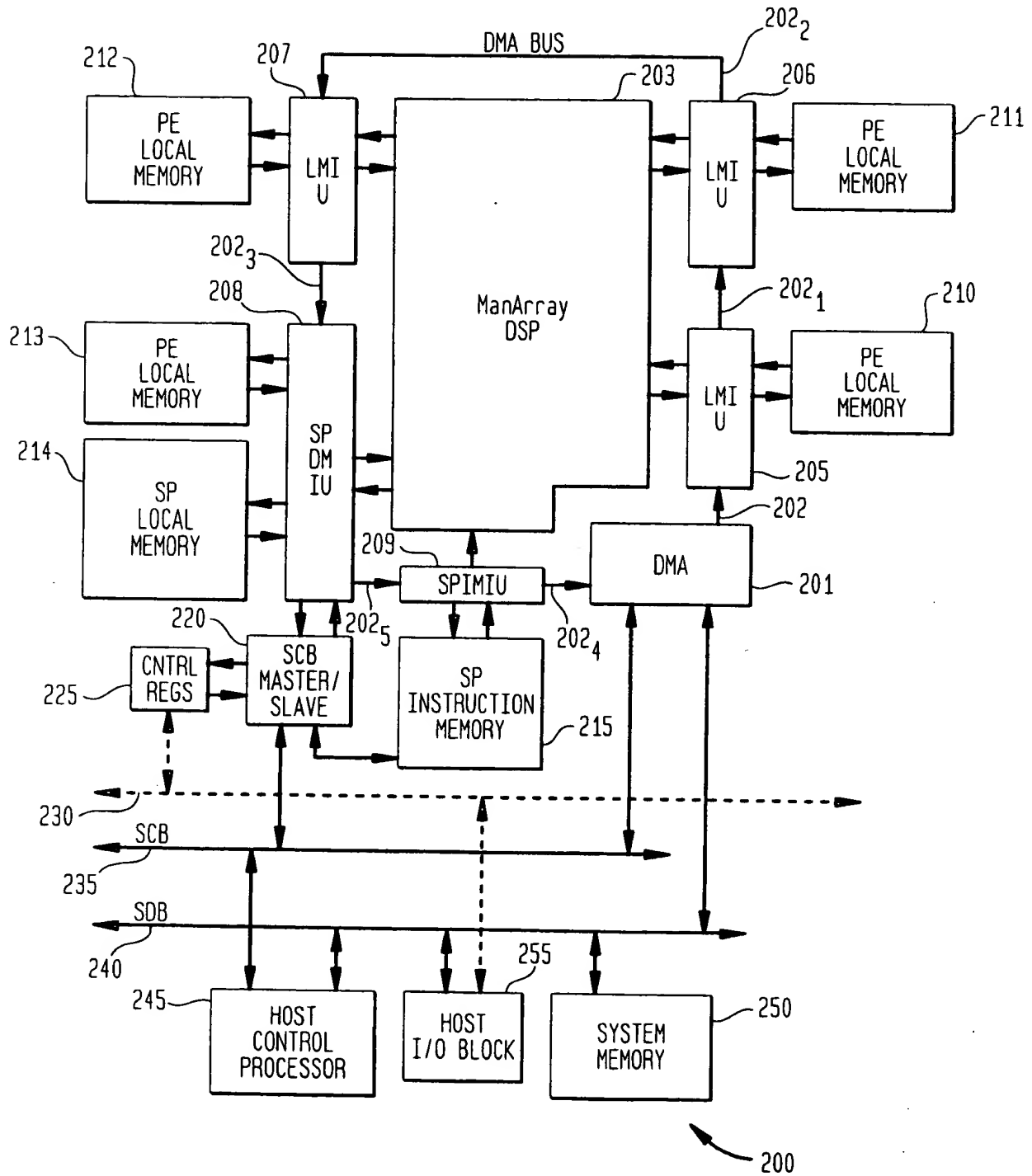


FIG. 3

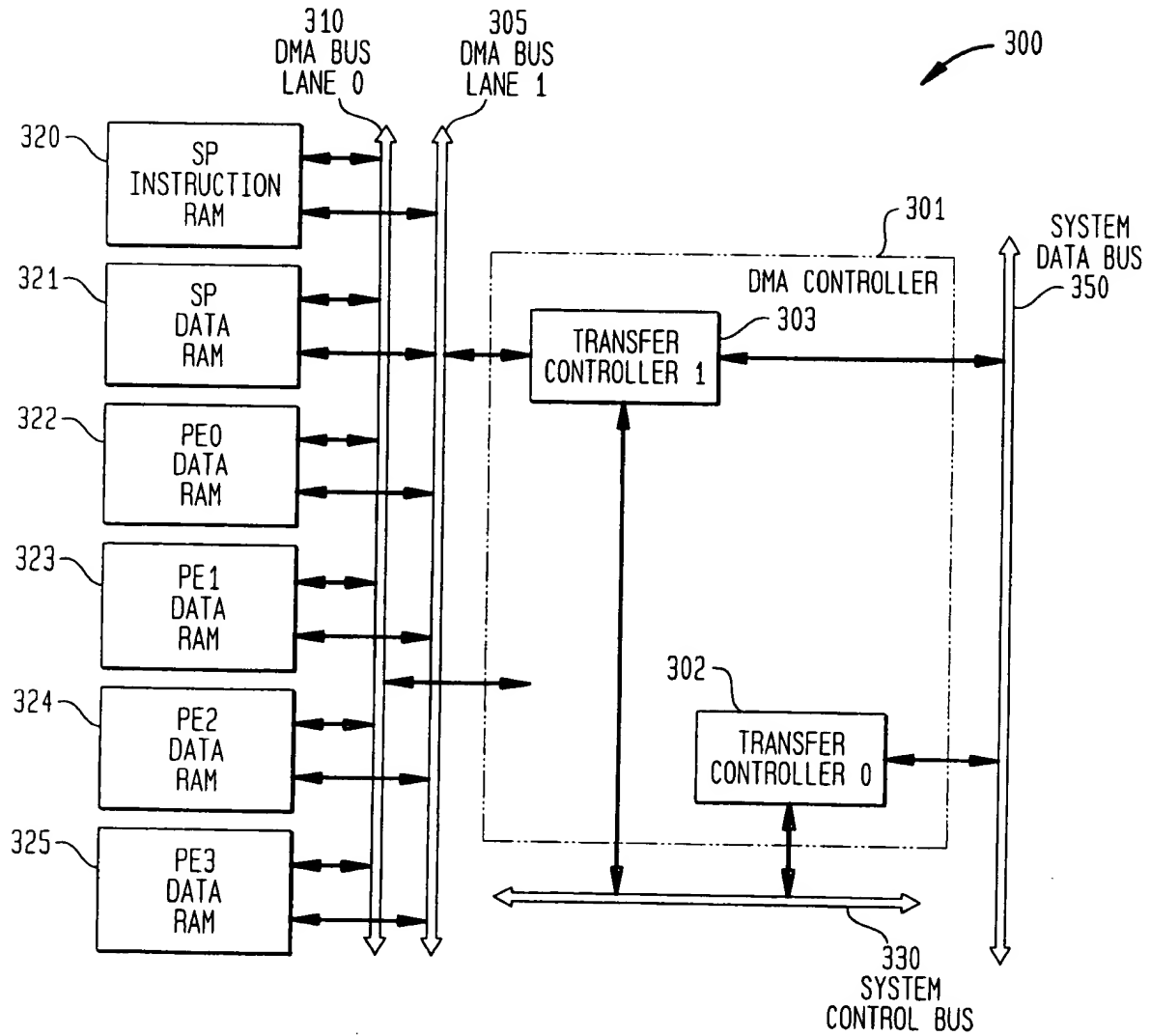


FIG. 4

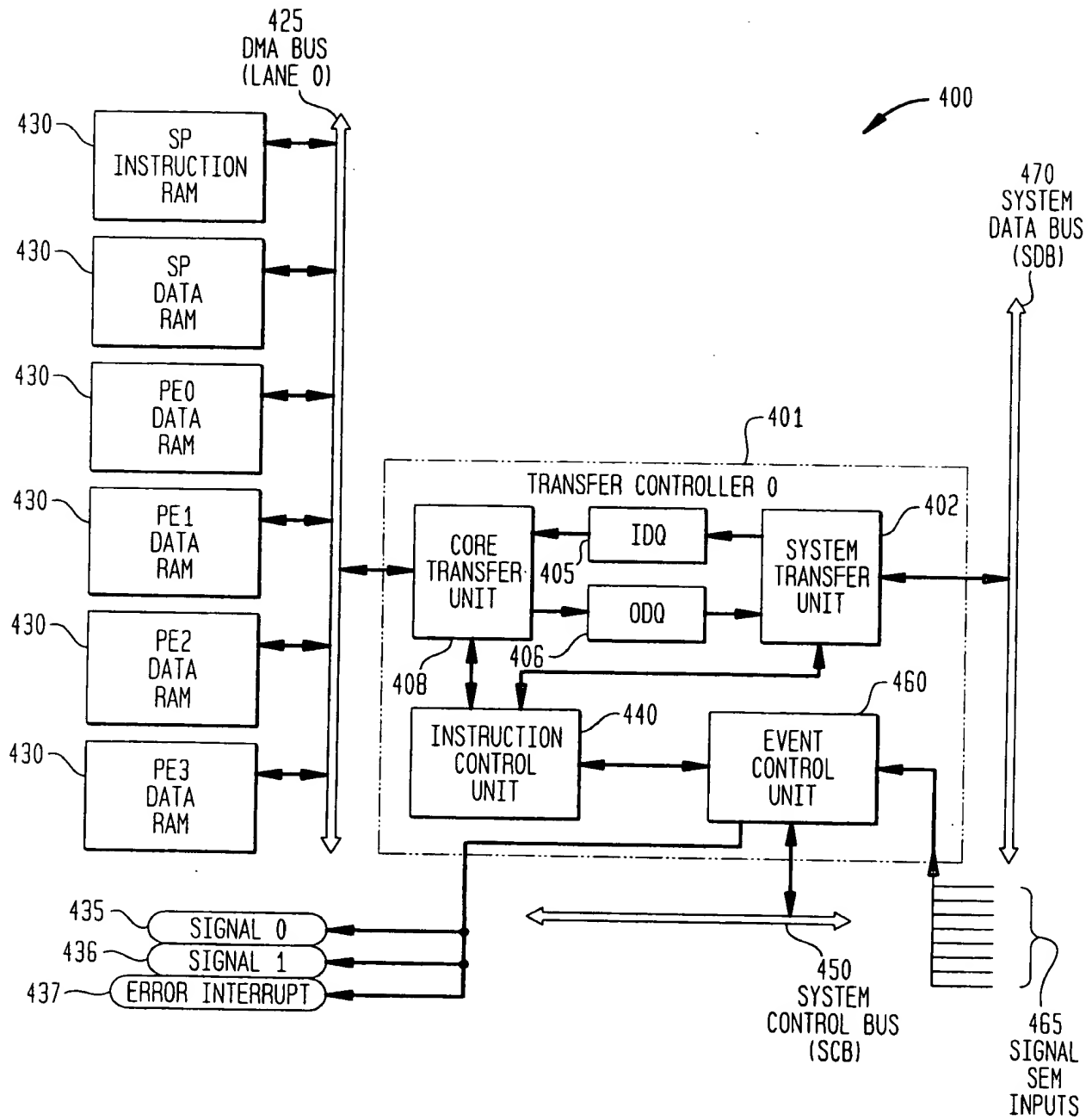


FIG. 5

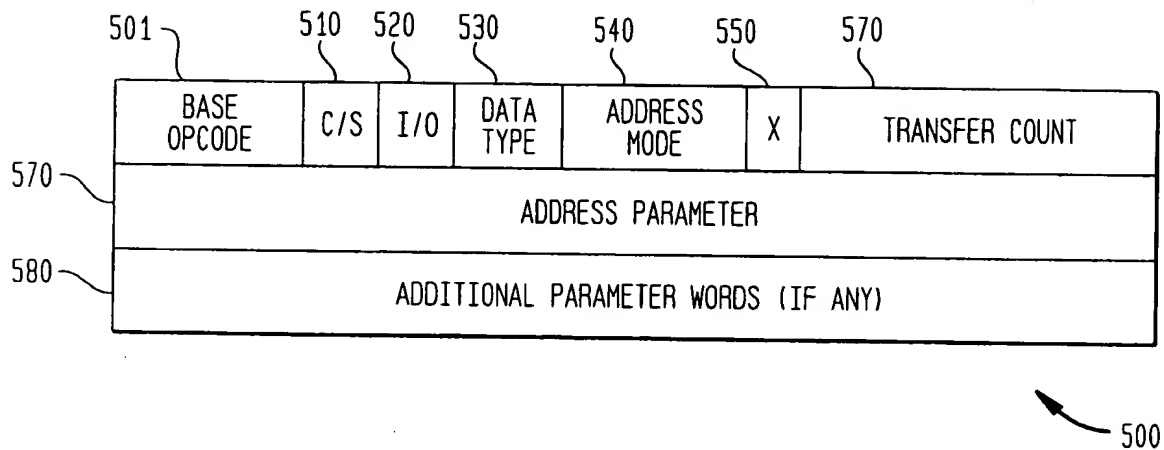


FIG. 6

Diagram 600 shows a mapping table between Virtual PE ID and Physical PE ID. The table has two columns: VIRTUAL PE ID (602) and PHYSICAL PE ID (604). The mapping is as follows:

602 VIRTUAL PE ID	604 PHYSICAL PE ID
0	1
1	2
2	3
3	0

600

FIG. 7

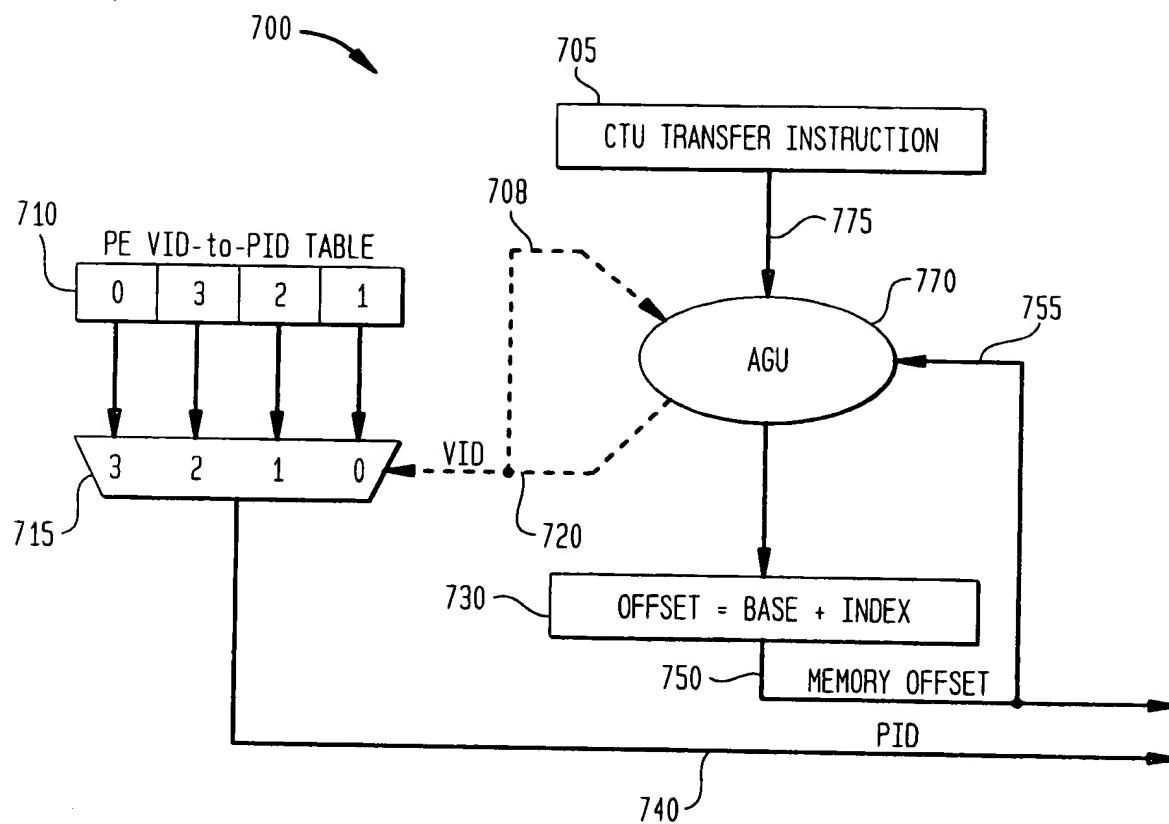


FIG. 8

800

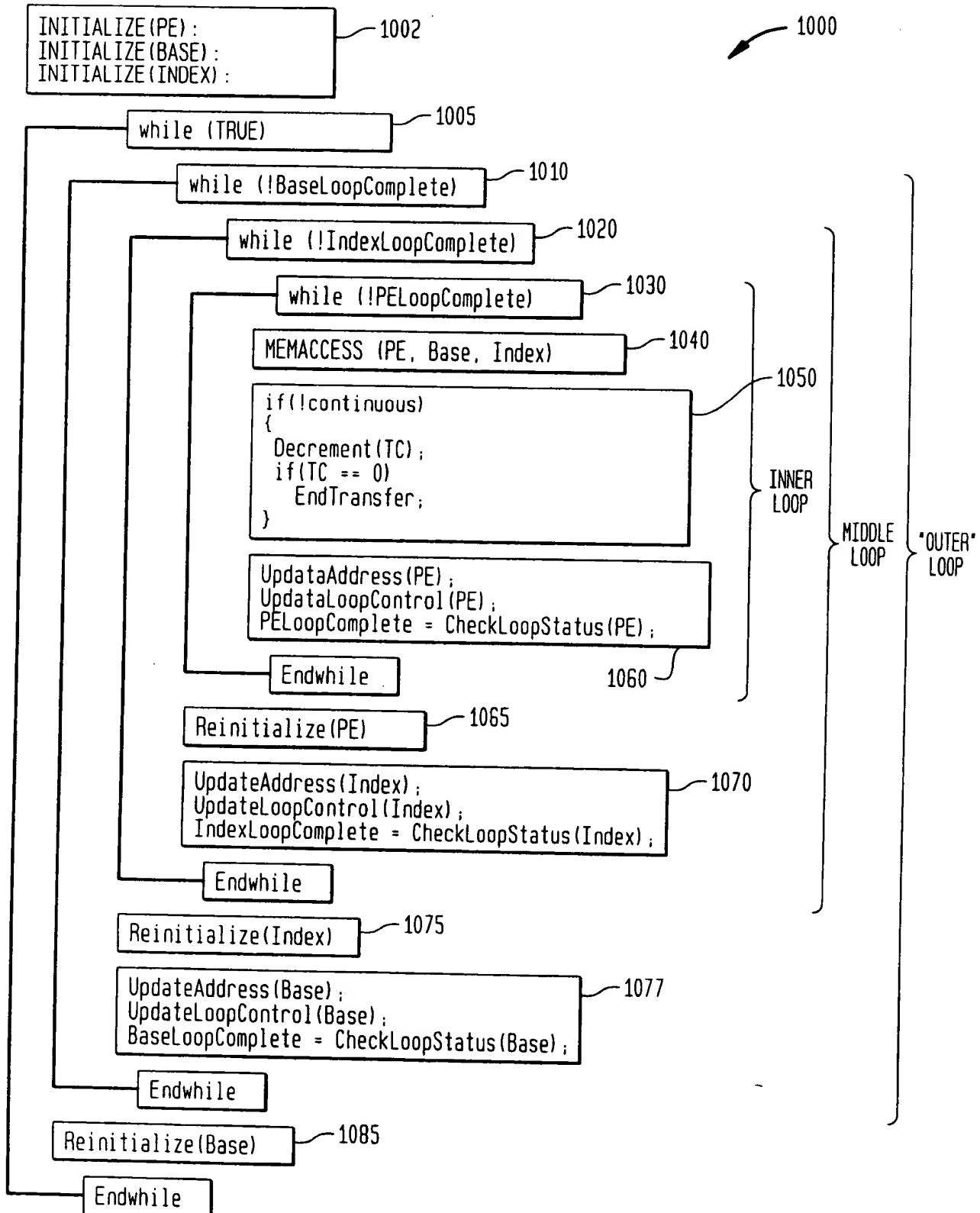
3 1	3 0	2 9	2 8	2 7	2 6	2 5	2 4	2 3	2 2	2 1	2 0	1 9	1 8	1 7	1 6	1 5	1 4	1 3	1 2	1 1	1 0	0 9	0 8	0 7	0 6	0 5	0 4	0 3	0 2	0 1	0 0
00		0110		MA TYPE 01		(USED FOR 2x4 TRANSLATE TABLE)																2x2 TABLE									
(USED FOR 4x4 TRANSLATE TABLE)																															
2x2 TABLE				CONTAINS A TABLE OF TWO BIT PE IDs. A SEQUENCE OF TWO BIT VALUES (STARTING WITH 0) WHICH SPECIFY THE PE VID, ARE APPLIED AS AN INDICES INTO THIS TABLE WHEN ONE OF THE PE ADDRESSING MODES IS USED IN A TRANSFER INSTRUCTION. THE TRANSLATED VALUE IS THEN USED TO PERFORM THE MEMORY ACCESS. WITH THIS APPROACH, PEs MAY BE ACCESSED IN ANY ORDER FOR THESE MODES.																											
MA TYPE				ManArray TYPE SPECIFIES THE CONFIGURATION TARGETED AND THEREFORE THE SIZE OF THE TABLE. 00 - 1x2 (UP TO 2 PEs) 01 - 2x2 (UP TO 4 PEs) 10 - 2x4 (UP TO 8 PEs) 11 - 4x4 (UP TO 16 PEs)																											

FIG. 9

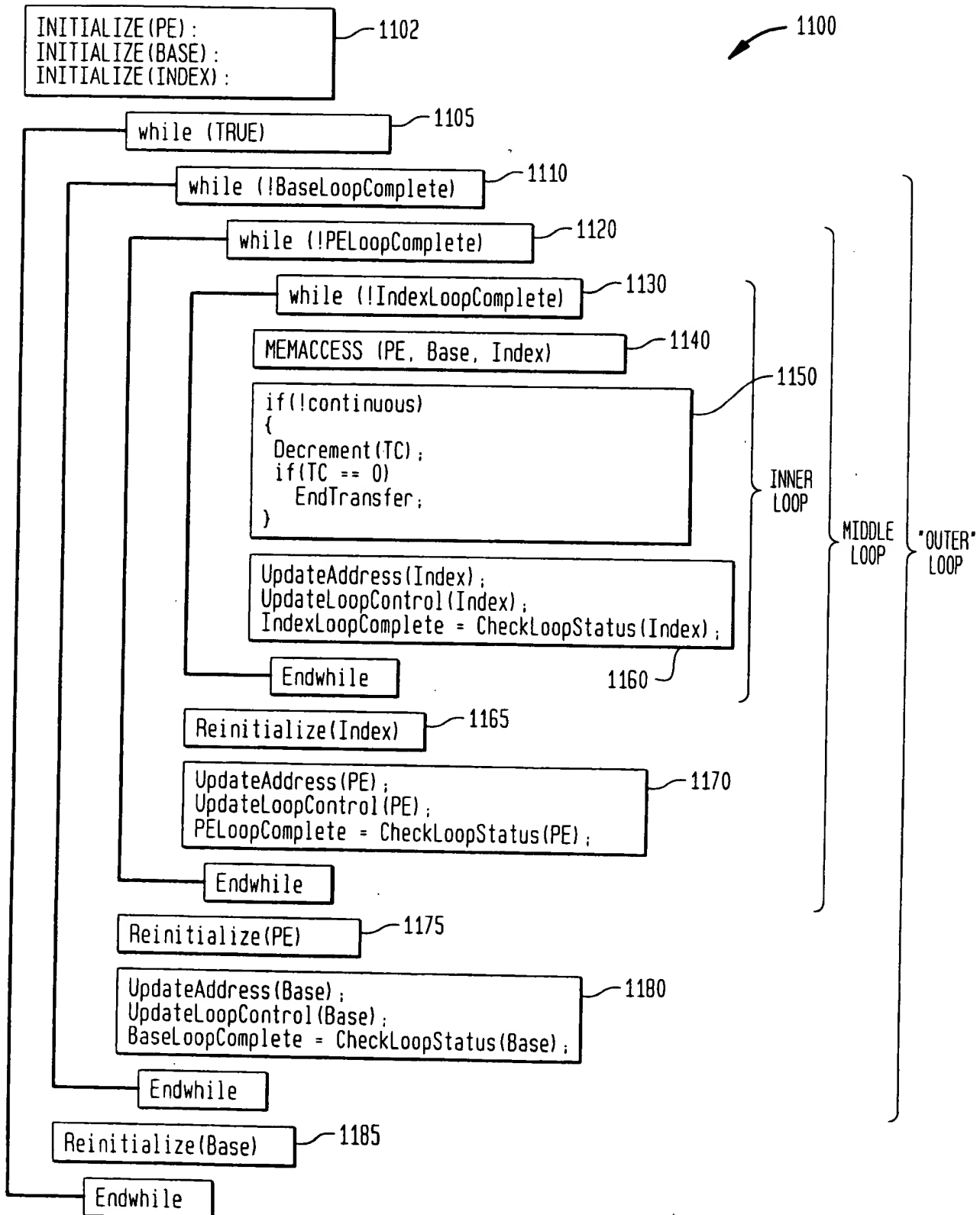
900

3 1	3 0	2 9	2 8	2 7	2 6	2 5	2 4	2 3	2 2	2 1	2 0	1 9	1 8	1 7	1 6	1 5	1 4	1 3	1 2	1 1	1 0	0 9	0 8	0 7	0 6	0 5	0 4	0 3	0 2	0 1	0 0
USED FOR PE ID TRANSLATION TABLES LARGER THAN 4 ELEMENTS																								PID3		PID2		PID1		PID0	

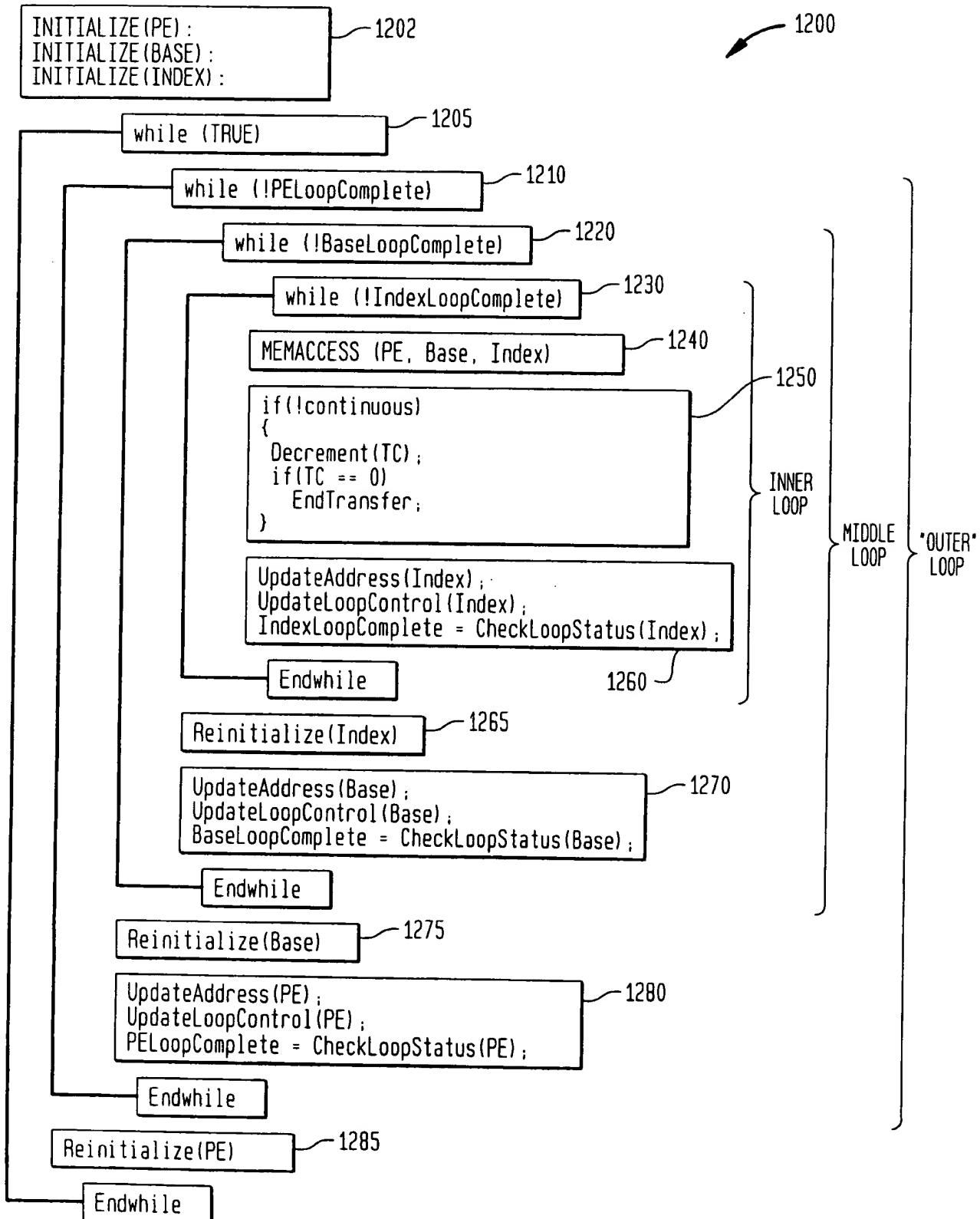
8/19
FIG. 10



9/19
FIG. 11



10/19
FIG. 12



12/19

FIG. 14

1400

LOOP CONTROL: BIP (PE ID VARIES FIRST, THEN INDEX, THEN BASE)				
ADDRESS	PE0	PE1	PE2	PE3
0x0000	0	1	2	3
0x0001				
0x0002	4	5	6	7
0x0003				
0x0004				
0x0005				
0x0006				
0x0007				
0x0008	8	9	10	11
0x0009				
0x000a	12	13	14	15

- AN INBOUND SEQUENCE OF 16 DATA ELEMENTS WITH VALUES 0,1,2,3,...15
- PETABLE SETTING OF 0x000000e4 (NO TRANSLATION OF PE IDs)
- TSI.block INSTRUCTION IN THE STU (READING THE 16 VALUES FROM SYSTEM MEMORY)
- TCI.blockcyclic INSTRUCTION IN THE CTU WITH PE COUNT = 4, LOOP CONTROL = BIP, BASE UPDATE = 8, BASE COUNT =, INDEX UPDATE = 2, INDEX COUNT = 2

FIG. 15

1500

LOOP CONTROL: BPI (INDEX VARIES FIRST, THEN PE ID, THEN BASE)				
ADDRESS	PE0	PE1	PE2	PE3
0x0000	0	2	4	6
0x0001				
0x0002	1	3	5	7
0x0003				
0x0004				
0x0005				
0x0006				
0x0007				
0x0008	8	10	12	14
0x0009				
0x000a	9	11	13	15

- AN INBOUND SEQUENCE OF 16 DATA ELEMENTS WITH VALUES 0,1,2,3,...15
- PETABLE SETTING OF 0x000000e4 (NO TRANSLATION OF PE IDs)
- TSI.block INSTRUCTION IN THE STU (READING THE 16 VALUES FROM SYSTEM MEMORY)
- TCI.blockcyclic INSTRUCTION IN THE CTU WITH PE COUNT = 4, LOOP CONTROL = BPI, BASE UPDATE = 8, BASE COUNT =, INDEX UPDATE = 2, INDEX COUNT = 2

FIG. 16

1600

LOOP CONTROL: PBI (INDEX VARIES FIRST, THEN BASE, THEN PE ID)				
ADDRESS	PE0	PE1	PE2	PE3
0x0000	0	4	8	12
0x0001				
0x0002	1	5	9	13
0x0003				
0x0004				
0x0005				
0x0006				
0x0007				
0x0008	2	6	10	14
0x0009				
0x000a	3	7	11	15

- AN INBOUND SEQUENCE OF 16 DATA ELEMENTS WITH VALUES 0,1,2,3,...15
- PETABLE SETTING OF 0x000000e4 (NO TRANSLATION OF PE IDs)
- TSI.block INSTRUCTION IN THE STU (READING THE 16 VALUES FROM SYSTEM MEMORY)
- TCI.blockcyclic INSTRUCTION IN THE CTU WITH PE COUNT = 4, LOOP CONTROL = BPI, BASE UPDATE = 8, BASE COUNT =, INDEX UPDATE = 2, INDEX COUNT = 2

NOTE THAT A FOR PBI MODE, THE BASE COUNT MUST BE 2 IN ORDER TO GET 2 "BLOCKS" OF DATA. INDEX COUNT CORRESPONDES TO THE NUMBER OF ELEMENTS WRITTEN BEFORE UPDATING THE NEXT ADDRESS VARIABLE. THE GAP BETWEEN ELEMENTS WITHIN A PE IS DUE TO THE INDEX UPDATE VALUE OF 2 (RATHER THAN 1)

3 1	3 0	2 9	2 8	2 7	2 6	2 5	2 4	2 3	2 2	2 1	2 0	1 9	1 8	1 7	1 6	1 5	1 4	1 3	1 2	1 1	1 0	0 9	0 8	0 7	0 6	0 5	0 4	0 3	0 2	0 1	0 0
CTU TRANSFER				I / 0	TYPE		PE SELECT INDEX		X	RSDV		CORE TRANSFER COUNT (CTC)																			
INDEX COUNT		RESERVED										STARTING TRANSFER ADDRESS (WITHIN PE MEMORY)																			
LOOP CTRL		INDEX COUNT		BASE UPDATE COUNT						BASE UPDATE (STRIDE)																					
IU7		IU6		IU5		IU4		IU3		IU2		IU1		IU0																	
<p>LOOP CTRL</p> <p>LOOP CTRL SPECIFIES A PARTICULAR ORDER IN WHICH PE, BASE AND INDEX VALUES ARE UPDATED. THREE POSSIBLE ORDERS ARE SELECTABLE WHICH CORRESPOND TO THREE ASSIGNMENTS OF PE, BASE AND INDEX UPDATE TO THREE NESTED CONTROL LOOPS (OUTER, MIDDLE AND INNER).</p> <p>00 - BASE (OUTER), INDEX (MIDDLE), PE (INNER) - BIP</p> <p>01 - BASE (OUTER), PE (MIDDLE), INDEX (INNER) - BPI</p> <p>10 - PE (OUTER), BASE (MIDDLE), INDEX (INNER) - PBI</p>																															
PE COUNT			<p>SPECIFIES THE NUMBER OF PEs TO BE ACCESSED FOR EACH TIME THE PE COUNTER IS SIGNALLED TO RELOAD. VALID VALUES ARE:</p> <p>0000 - MAX NUMBER OF PEs AS SPECIFIED IN THE PE CONFIGURATION REGISTER</p> <p>0001 - 1</p> <p>0010 - 2</p> <p>0011 - 3 ETC., ETC.</p>																												
BASE UPDATE (STRIDE)			DISTANCE BETWEEN SUCCESSIVE BLOCKS. UNITS ARE OF "DATA TYPE" SIZE.																												
BASE UPDATE COUNT			USED FOR PBI LOOP CONTROL. SPECIFIES THE NUMBER OF TIMES THE BASE IS UPDATED BEFORE EXITING TO THE OUTER LOOP (PE UPDATE). RANGE IS 1 TO 256.																												
IUx			IU0 - IU7 FORM AN INDEX UPDATE TABLE WITH EACH ENTRY BEING A 4-BIT UPDATE VALUE. UPDATE VALUES ARE INTEGERS IN THE RANGE OF -8 TO +7																												
INDEX COUNT			NUMBER OF TIMES TO EXECUTE THE INDEX UPDATE LOOP. THIS VARIABLE PROVIDES THE LOOP EXIT CONTROL FOR THE INDEX LOOP.																												

FIG. 18

1800

LOOP CONTROL: BIP (INDEX VARIES FIRST, THEN BASE, THEN PE ID)				
ADDRESS	PE0	PE1	PE2	PE3
0x0000	0	1	2	3
0x0001	24	25	26	27
0x0002	4	5	6	7
0x0003	20	21	22	23
0x0004	8	9	10	11
0x0005	16	17	18	19
0x0006	12	13	14	15
0x0007				
0x0008	28	29	30	31
0x0009				
0x000a	32	33	34	35

PATTERN ABOVE RESULTS FROM AFTER A TRANSFER WITH THE FOLLOWING ASSUMPTIONS:

- TSI.block INSTRUCTION READS SUCCESSIVE ADDRESSES FROM SYSTEM MEMORY, DATA ELEMENT VALUES ARE 0,1,2,...etc.
- TCI.select INDEX INSTRUCTION PLACES VALUES IN PE MEMORIES USING THE FOLLOWING PARAMETERS
- ASSUME NO PE VID-to-PID TRANSLATION
- TRANSFER COUNT = 36
- PE ADDRESS = 0
- PE COUNT = 4
- LOOP CONTROL = BIP
- BASE UPDATE COUNT = 0
- BASE UPDATE = 8
- INDEX UPDATE TABLE VALUE IS 0x00EEF222 WHICH GIVES UPDATES 2,2,2,-1,-2,-2
- INDEX COUNT = 7

FIG. 20

2000

LOOP CONTROL: BIP (INDEX VARIES FIRST, THEN BASE, THEN PE ID)				
ADDRESS (WORDS)	PE0	PE1	PE2	PE3
0x0000		0	1	2
0x0001		3	4	5
0x0002	9	6	7	8
0x0003		10	11	12
0x0004				
0x0005				
0x0006				
0x0007				
0x0008		13	14	15
0x0009		16	17	18
0x000a	22	19	20	21
0x000a		23	24	25

PATTERN ABOVE RESULTS FROM AFTER A TRANSFER WITH THE FOLLOWING ASSUMPTIONS:

- TSI.block INSTRUCTION READS SUCCESSIVE ADDRESSES FROM SYSTEM MEMORY. DATA ELEMENT VALUES ARE 0,1,2,...etc.
- ASSUME PE TRANSLATE TABLE MAPS 0→1, 1→2, 2→3, 3→0
- TCI.selectpe INSTRUCTION PLACES VALUES IN PE MEMORIES USING THE FOLLOWING PARAMETERS
- TRANSFER COUNT = 26
- INITIAL PE ADDRESS OFFSET = 0
- PE COUNT = NOT USED
- LOOP CONTROL = BIP
- BASE UPDATE COUNT = 0
- BASE UPDATE = 8
- INDEX UPDATE = 1
- INDEX COUNT = 4
- PE TABLE IS 0x00000F77
 - FIRST PASS SELECT VIDs: 0, 1, 2 (TRANSLATION CONVERTS THESE TO PIDs: 1,2,3)
 - NEXT PASS SELECT VIDs 0,1,2 (TRANSLATION CONVERTS THESE TO PIDs: 1,2,3)
 - NEXT PASS SELECT VIDs 0,1,2,3 (TRANSLATION CONVERTS THESE TO PIDs: 1,2,3,0)

FIG. 22

2200

LOOP CONTROL: BIP (INDEX VARIES FIRST, THEN BASE, THEN PE ID)				
ADDRESS (WORDS)	PE0	PE1	PE2	PE3
0x0000		0	1	2
0x0001				
0x0002		3	4	5
0x0003				
0x0004				
0x0005	9	6	7	8
0x0006		10	11	12
0x0007				
0x0008		13	14	15
0x0009				
0x000a				
0x000a	19	16	17	18

PATTERN ABOVE RESULTS FROM AFTER A TRANSFER WITH THE FOLLOWING ASSUMPTIONS:

- TSI.block INSTRUCTION READS SUCCESSIVE ADDRESSES FROM SYSTEM MEMORY, DATA ELEMENT VALUES ARE 0, 1, 2, ... etc.
- ASSUME PE TRANSLATE TABLE MAPS 0 → 1, 1 → 2, 2 → 3, 3 → 0
- TCI.selectpe INSTRUCTION PLACES VALUES IN PE MEMORIES USING THE FOLLOWING PARAMETERS
- TRANSFER COUNT = 20
- INITIAL PE ADDRESS OFFSET = 0
- PE COUNT = NOT USED
- LOOP CONTROL = BIP
- BASE UPDATE COUNT = 0
- BASE UPDATE = 6
- INDEX COUNT = 3
- INDEX TABLE = 0x00000032 (+2, THEN +3)
- PE HELPE IS 0x00000F77
 - FIRST PASS SELECT VIDs 0, 1, 2 (TRANSLATION CONVERTS THESE TO PIDs: 1, 2, 3)
 - NEXT PASS SELECT VIDs 0, 1, 2 (TRANSLATION CONVERTS THESE TO PIDs: 1, 2, 3)
 - NEXT PASS SELECT VIDs 0, 1, 2, 3 (TRANSLATION CONVERTS THESE TO PIDs: 1, 2, 3, 0)